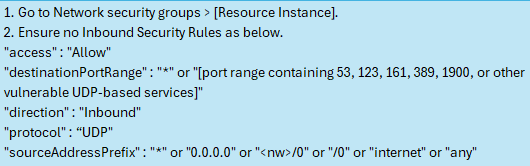
**Azure Policy**

Please refer to [Azure Policy definition structure basics](https://learn.microsoft.com/en-us/azure/governance/policy/concepts/definition-structure-basics) to understand about the resource compliance conditions & effects. Implement & walkthrough a custom policy to meet the following requirement.

* 1. Ensure UDP access from Internet is blocked in the NSG rules by default, as follows.



1. Create a resource group called **Alpha2phi**
2. From Azure Policy, create a policy using [policy.json](https://github.com/alpha2phi/azure-techn-q/blob/main/block-udp/policy.json)
3. Assign the policy to the **Alpha2phi** resource group
4. From Network security groups, create NSG rules in **Alpha2phi** resource group.

A screenshot of a computer

Description automatically generated

A screenshot of a computer error message

Description automatically generated

**Sample Scenarios**

|  |  |
| --- | --- |
| Configurable | Evaluation |
| Protocol: UDP  Destination Ports: 53 | Blocked |
| Protocol: Any  Destination Port: 123 | Blocked |
| Protocol: Any or UDP  Destination Ports: 53,123,8080 | Blocked |
| Protocol: Any or UDP  Destination Ports: 53,123,1000-2000 | Blocked |
| Protocol: Any or UDP  Destination Ports: 8080,8081 | Allowed |
| Protocol: Any or UDP  Destination Ports: 50-60 | Blocked |
| Source: Internet  Destination Ports: Any  Protocol: Any | Blocked |
|  |  |

**Limitation**

* Currently for destination port ranges with “-“, the rule is restrictive as it blocks when it detects there is a “-“ in the input, without checking the UDP ports. To specify port ranges, use comma-separated values instead.
* Need to modify the rule to allow flexibility to specify port ranges using “-“with validation of vulnerable UDP ports. E.g. below is for a single port 53. Need to investigate how to adapt it to validate against an array of ports. A better way could be using “initiative” with different UDP policies.

{

"value": "[if(and(not(empty(field('Microsoft.Network/networkSecurityGroups/securityRules/destinationPortRange'))), contains(field('Microsoft.Network/networkSecurityGroups/securityRules/destinationPortRange'),'-')), and(lessOrEquals(int(first(split(field('Microsoft.Network/networkSecurityGroups/securityRules/destinationPortRange'), '-'))),53),greaterOrEquals(int(last(split(field('Microsoft.Network/networkSecurityGroups/securityRules/destinationPortRange'), '-'))),53)), 'false')]",

"equals": "true"

},

{

"count": {

"field": "Microsoft.Network/networkSecurityGroups/securityRules/destinationPortRanges[\*]",

"where": {

"value": "[if(and(not(empty(first(field('Microsoft.Network/networkSecurityGroups/securityRules/destinationPortRanges[\*]')))), contains(first(field('Microsoft.Network/networkSecurityGroups/securityRules/destinationPortRanges[\*]')),'-')), and(lessOrEquals(int(first(split(first(field('Microsoft.Network/networkSecurityGroups/securityRules/destinationPortRanges[\*]')), '-'))),53),greaterOrEquals(int(last(split(first(field('Microsoft.Network/networkSecurityGroups/securityRules/destinationPortRanges[\*]')), '-'))),53)) , 'false')]",

"equals": "true"

}

},

"greater": 0

},

**Improvements**

* Use remediation task to fix existing rules
* Instead of a single custom policy definition, leverage “initiative” to group different policies to block UDP traffic. This makes it easier to group and assign related UDP policies, and simpler to create different policies for specific UDP traffic like DNS, LDAP, etc.

**Resource Graph Query**

Please refer to [Advanced Resource Graph query samples](https://learn.microsoft.com/en-us/azure/governance/resource-graph/samples/advanced?tabs=azure-cli). Write a Resource Graph query which summarizes the count of Storage Accounts been encrypted with CMK grouped by BusinessUnit tag (BusinessUnit tag only present at Resource Group level).

1. Create the resource groups and storage accounts for testing

|  |  |  |
| --- | --- | --- |
| Resource | Type | Remarks |
| Alpha2phi | Resource group | Tag: BusinessUnit = Finance |
| Alpha2phi\_2 | Resource group | Tag: BusinessUnit = HR |
| Alpha2phi\_3 | Resource group | No tag |
| storagecmk168 | Storage account | * CMK encrypted * Under Alpha2phi |
| storagecmk1682 | Storage account | * CMK encrypted * Under Alpha2phi\_2 |
| storagecmk1683 | Storage account | * CMK encrypted * Under Alpha2phi\_3 |
| storagecmk16834 | Storage account | * CMK encrypted * Under Alpha2phi |
| storagenocmk168 | Storage account | * Microsoft managed encryption * Under Alpha2phi |

1. From Azure portal Resource graph explorer, run the [storage\_account.kql](https://github.com/alpha2phi/azure-techn-q/blob/main/query/storage_acct.kql) query.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**PowerShell Script**

1. Implement a Powershell script to set all Azure Key Vault’s keys & secrets to expire in 2 years’ time.
2. Make sure we have [Azure PowerShell](https://learn.microsoft.com/en-us/powershell/azure/install-azure-powershell?view=azps-13.0.0) installed.
3. Create a keyvault with keys, secrets, and certificates for testing.

A screen shot of a computer

Description automatically generated

1. Run the [set\_keyvault\_expiry.ps1](https://github.com/alpha2phi/azure-techn-q/blob/main/script/set_keyvault_expiry.ps1) script.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

* Disabled keys and secrets are skipped
* Keys and secrets related to certificates are skipped

**Improvements**

* Add in error handling
* Tune the performance if there are a lot of key values, secrets and keys to change
* Refactor the code to make it more modular

***Notes***

Code repository is available here - [alpha2phi/azure-techn-q: Azure technical questions](https://github.com/alpha2phi/azure-techn-q)

***Disclaimer****: Scripts and code generated with the help of ChatGPT.*